Universidad Andrés Bello
Facultad de Ingeniería
Ingeniería en Computación e Informática

# Estructura de Datos
# Solemne II

**Nombre**: _____ **Nota:** _____

**Profesores:** Carlos Contreras Bolton – José Luis Allende – Felipe Reyes González
**Ayudantes:** Carlos Rey – Daniela Ubilla – Tamara Saéz **Fecha:** 29 de Mayo del 2014

**Instrucciones:**

- Coloque su nombre a todas las hojas.
- Apague o silencie sus celulares. **NO** se podrá contestar llamadas ni visualizar el celular. Si se realiza una de las acciones anteriores obtendrá nota mínima.
- Toda copia o intento de copia será calificada con nota mínima.
- Tiene 90 minutos para realizar la prueba.
- El ejercicio 5 es **obligatorio**. Del resto de ejercicios escoja 3 para desarrollar.

| | |
|---|---|
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| Suma: | |
| | +10 |
| Nota: | |

## Pregunta 1 (15 puntos)

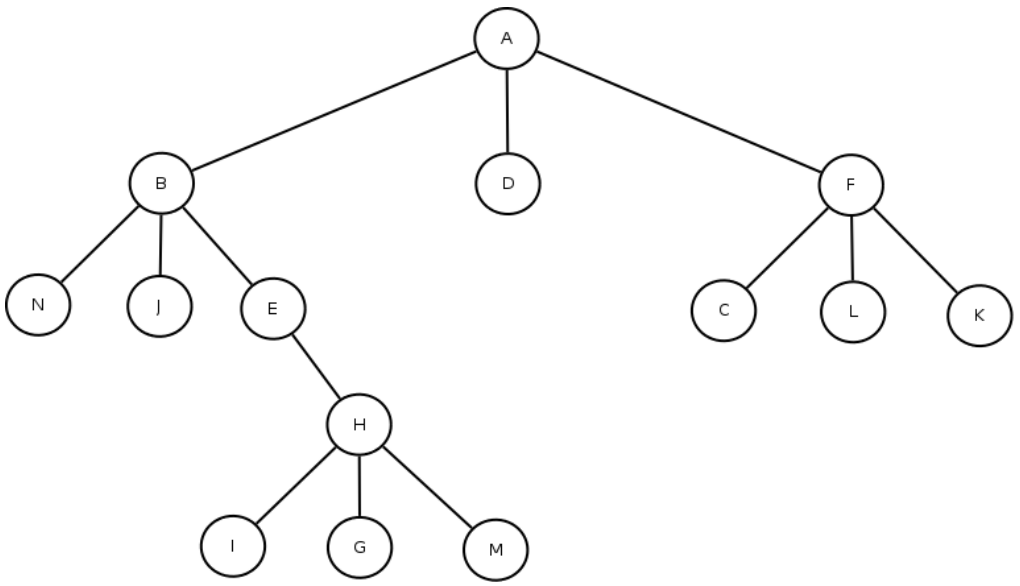
Para los siguientes recorrido:

**Inorden** N B J E I H G M A D C F L K

**Postorden** N J I G M H E B D C L K F A

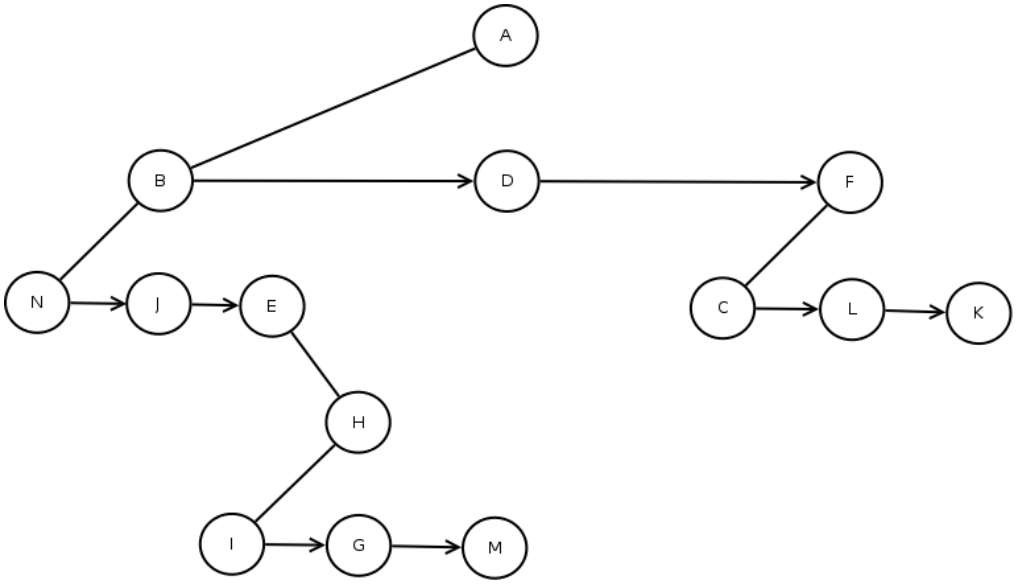
**Preorden** A B N J E H I G M D F C L K

Encuentre el árbol e indique:




1. Altura del árbol: 4
2. Altura de `F`: 1

3. Profundidad de `D`: 1
4. Grado del árbol: 3
5. Grado de `L`: 0
6. Su representación *hijo izquierdo hermano derecho*.

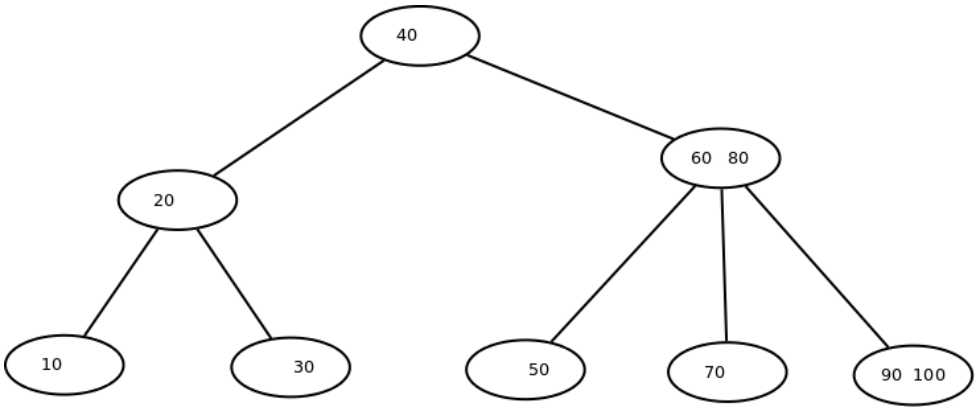



## Pregunta 2 (15 puntos)

Dada la siguiente secuencia de números

$$50, 60, 70, 40, 30, 20, 10, 80, 90, 100$$

Genere el árbol 2-3 correspondiente.




## Pregunta 3 (15 puntos)

Suponga la siguiente lista de números:

$$13, 7, 21, 15, 27, 18, 4, 11, 30, 31, 29, 12$$

Insertelos en un árbol de tipo AVL e indique la cantidad de nodos que se deben visitar para encontrar el número 11. Luego, elimine los números $4, 21, 15, 31$ y muestre el resultado preorden.

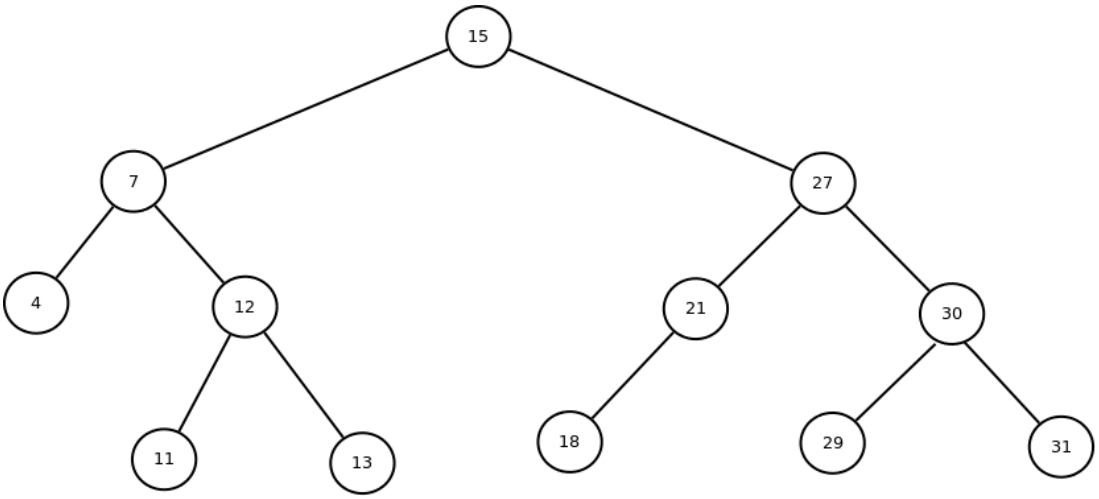



Figura 1: Se precisa visitar 4 nodos hasta llegar al número 11.

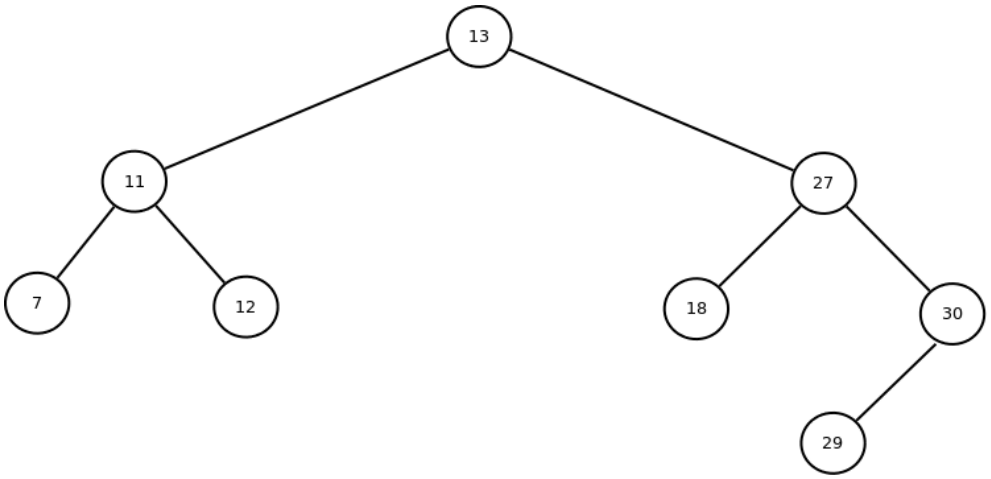



Figura 2: Eliminados $4, 21, 15, 31$.

PREORDEN:

$13, 11, 7, 12, 27, 18, 30, 29$

## Pregunta 4 (15 puntos)

El algoritmo de ordenamiento conocido como `heapsort` se basa en la idea de ordenar un conjunto de datos según las propiedades definidas por los heap binarios. Dado el conjunto de datos siguiente:

$$13, 7, 21, 15, 27, 18, 4, 11, 30, 31, 29, 12$$

Realice paso a paso el procedimiento definido para `heapsort`.

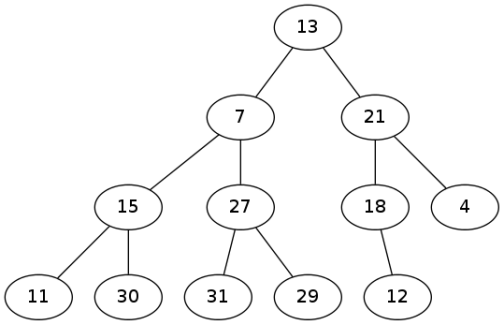



Figura 3: Traspasar directo desde el arreglo al heap.

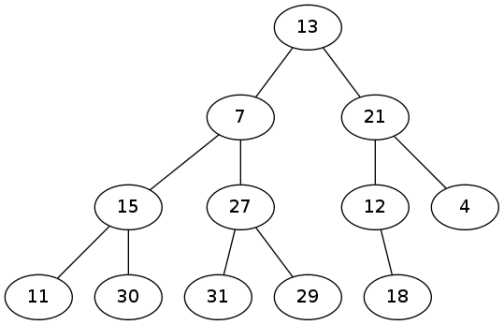



Figura 4: Primera parte, Hundir(18).

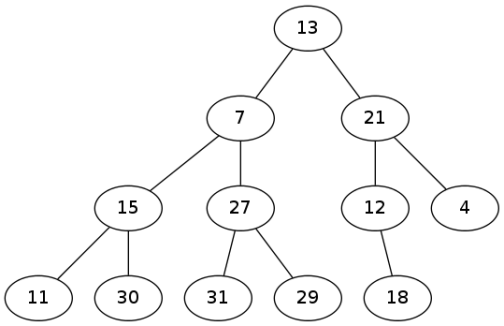



Figura 5: Hundir(27).

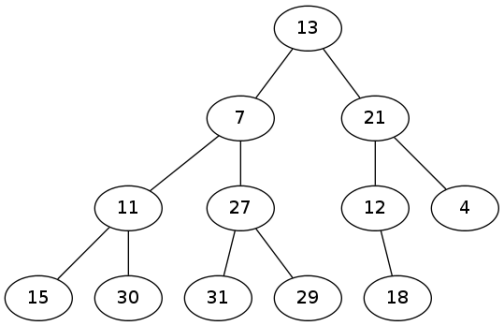



Figura 6: Hundir(15).

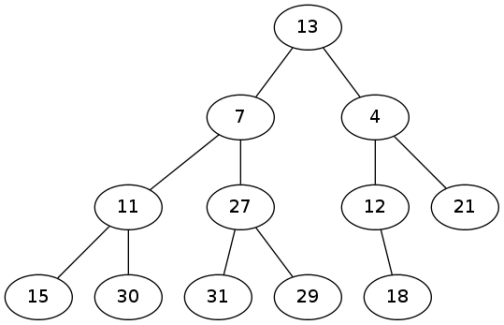



Figura 7: Hundir(21).

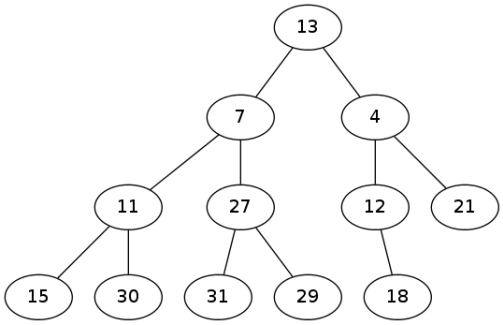



Figura 8: Hundir(7).

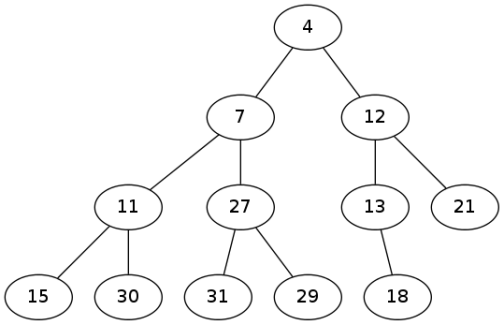



Figura 9: Hundir(13).

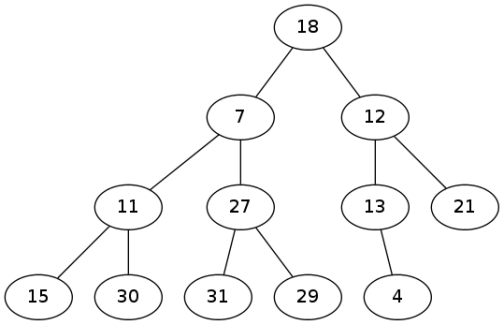



Figura 10: Segunda parte, Intercambiar(4, 18).

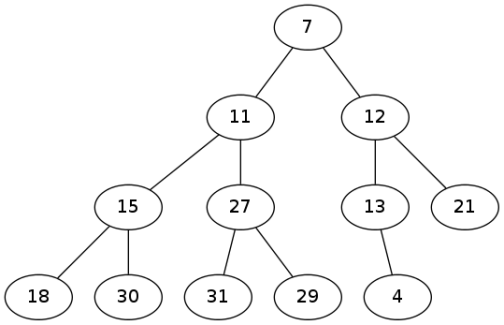



Figura 11: Hundir(18).

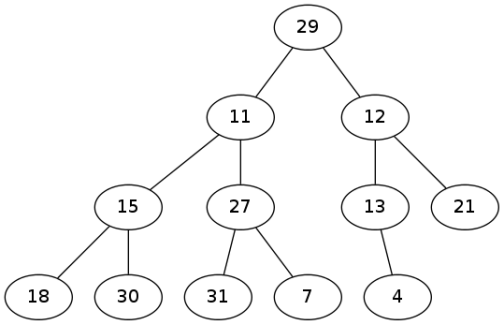



Figura 12: Intercambiar(7, 29).

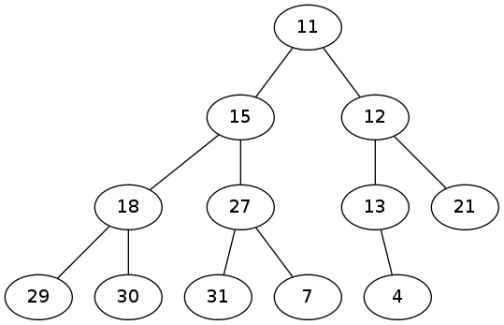



Figura 13: Hundir(29).

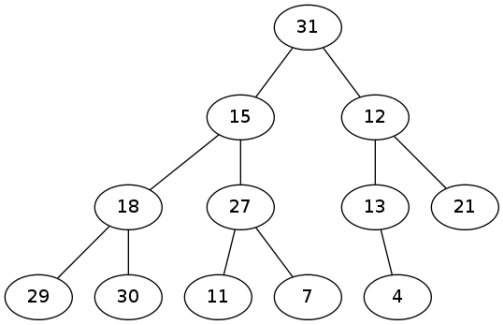



Figura 14: Intercambiar(11, 31).

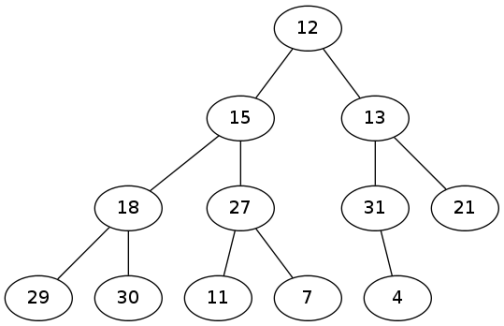



Figura 15: Hundir(31).

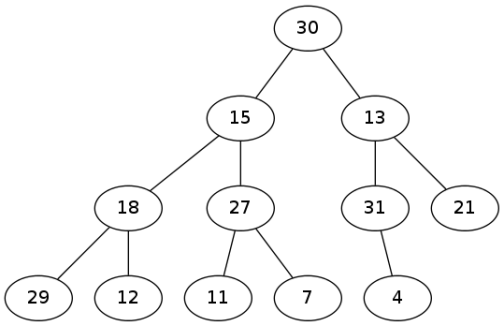



Figura 16: Intercambiar(12, 30).

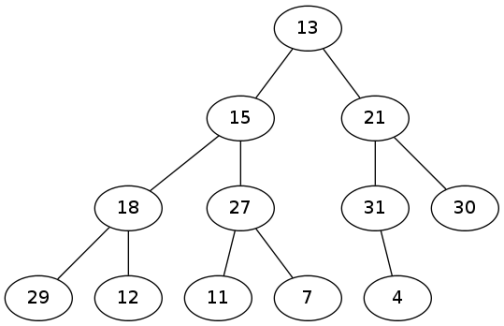



Figura 17: Hundir(30).

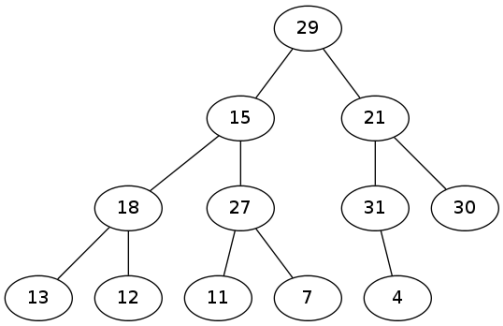



Figura 18: Intercambiar(13, 29).

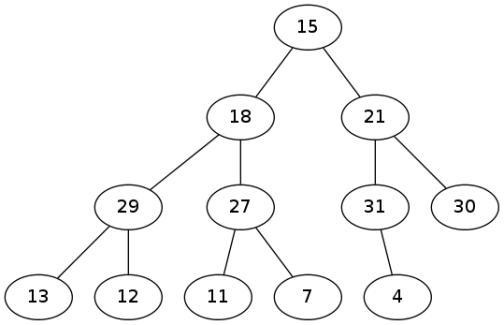



Figura 19: Hundir(29).

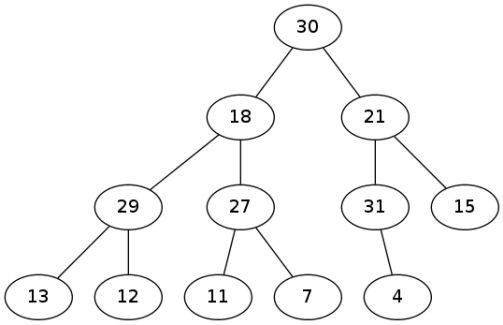



Figura 20: Intercambiar(15, 30).

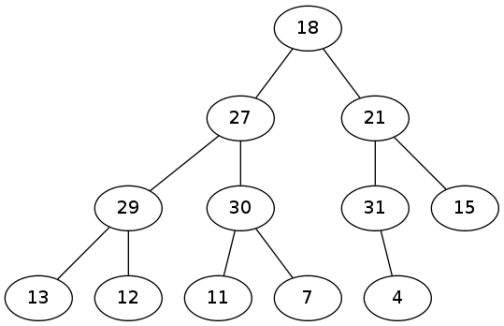



Figura 21: Hundir(30).

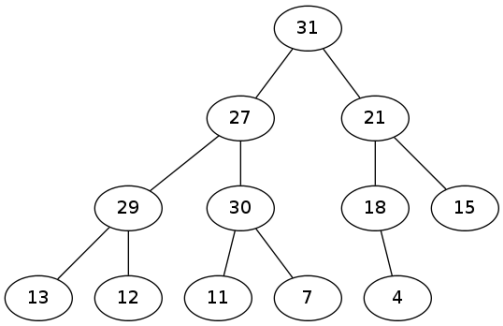



Figura 22: Intercambiar(18, 31).

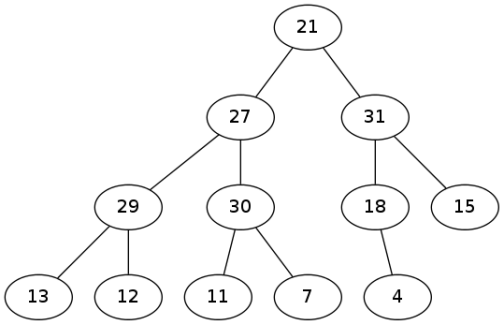



Figura 23: Hundir(31).

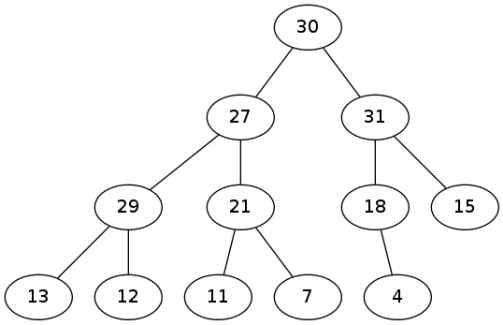



Figura 24: Intercambiar(21, 30).

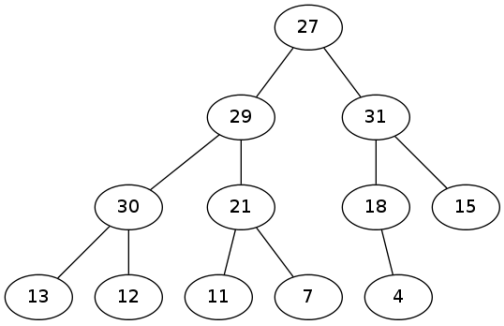



Figura 25: Hundir(30).

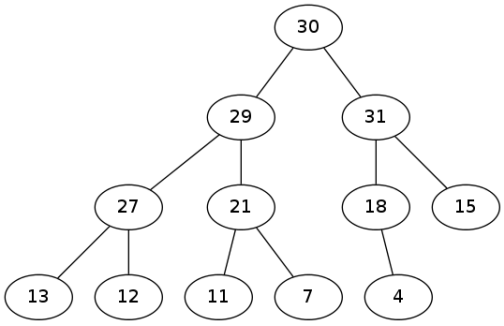



Figura 26: Intercambiar(27, 30).

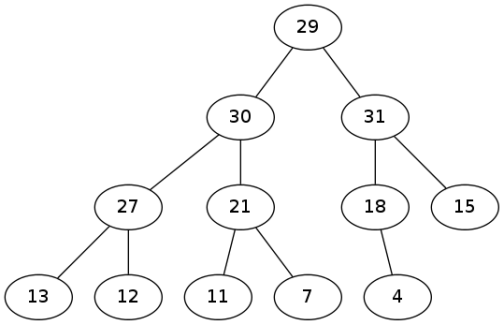



Figura 27: Hundir(30).

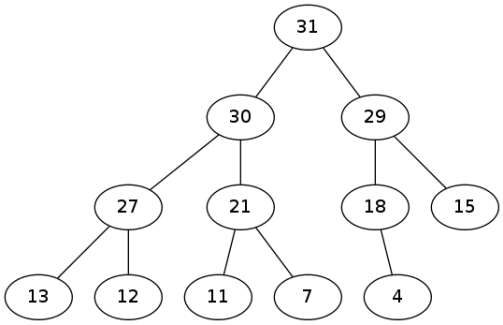



Figura 28: Intercambiar(29, 31).

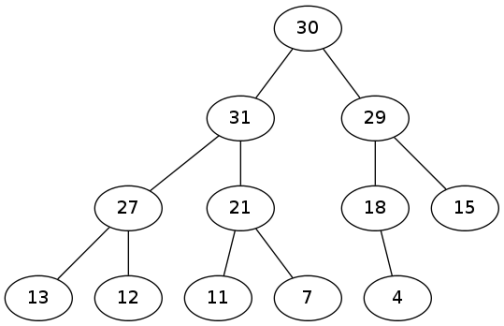



Figura 29: Hundir(31).

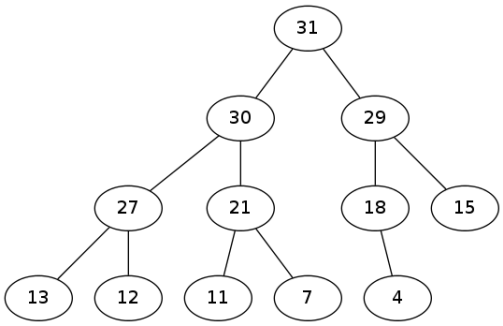



Figura 30: Intercambiar(30, 31) y Final.

## Pregunta 5 (15 puntos)

Escriba una función que cuente los nodos internos (sin considerar la raíz) con valores entre dos números dados (inclusive) y retorne ese valor.

```
1  int contarInter(Nodo *a, int n1, int n2)
2  {
3    int contar_flag = 1;
4    if(a == NULL)
5      return 0;
6    if((a->izq == NULL) && (a->der == NULL)) /* no contar hojas */
7      return 0;
8    if((a->dato < n1) || (a->dato > n2)) /* contar nodos en rango */
9      contar_flag = 0;
10   /* visitar arbol completo correctamente */
11   return contar_flag + contarInter(a->izq, n1, n2) + contarInter(a->der, n1, n2);
12 }
13
14 int contarInternos(Nodo *a, int n1, int n2)
15 {
16   int total = 0;
17   total += contarInter(a->izq, n1, n2); /* no contar raíz */
18   total += contarInter(a->der, n1, n2);
19   return total;
20 }
```